Application No. 09/823,991
Amendment dated: February 10, 2004
Reply to Office Action dated November 10, 2003

**Amendments to the Specification**

Please replace the paragraphs beginning at p. 5, line 27 and ending at p. 14, line 25, with the following amended paragraphs:

Referring to **Figure 2**, the software architecture of an embodiment the present invention has three layers, namely, destinations, people or network nodes, and devices. Entities of each layer are represented by agents **110** linked by relationships **130**. Specifically, as illustrated in **Figure 17**, there are destination agents **112**, person or node agents **114**, and device agents **116**. It should, however, be noted that the present invention is not restricted to three layers and that fewer or more layers are also possible in alternative systems. The three layer system described herein is especially applicable to a PBX replacement. However, a system can be designed to have two layers, or more than three, if desired. There can also be sub-layers within layers. Furthermore, entities are software constructs used by the system to establish communication paths. In the present example, it is convenient to associate entities with agents that correspond to real world counterparts. This is not, however, a necessary requirement of the architecture and the invention includes the possibility of other types of entities that may be entirely internal to the system.

Referring to **Figure 3**, relationships **130** between agents **110** are defined by policies **120**. A series of policies **120** forms a policy chain **126**. In a communication, a communication path is established by the use of these agents. Specifically, a communication path from a first agent to a second agent consists of the communication link between the first and second agents. If there are one or more intermediate agents then the communication path consists of the links between the first link to the one or more intermediate agents, any links between the one or more intermediate agents and ~~the~~ between the one or more intermediate agents to the second agent. The communication path established from the first agent to the second agent defines the attributes of the communication.

It is possible for two agents to be directly connected to each other. Indeed, this corresponds to the default case where "nothing special" happens. For example, a person node may be directly connected to a device if that person uses only one communications device, such as a landline office telephone with no voicemail, or other added features. In such a case, no additional processing logic is required and agents can be directly interconnected with no intervening policies. This direct connection is typical of conventional call control systems in which a phone number is tied to a device. Any changes to the system, for example to add increased functionality such as voicemail, conferencing, etc., require working around

this inherent coupling between a phone number and a device, and patching or adding to the system in a piecemeal fashion.

By contrast, the architecture of a system according to the present invention does not assume that a phone number is always associated with a person, or that a person is always associated with a particular device. Of course, such a direct association can be easily implemented in the present invention.

The present invention facilitates the addition of enhanced functionality by the use of agents to represent destinations, nodes, and devices. Simply adding policies **120** to the policy chain **126** that defines the relationship **130** between agents **110** increases functionality. Changes in functionality can be simply effected by suitably modifying the corresponding policies **120**. If the system is implemented using object-oriented programming, then changes to an object class can implement a consistent global set of changes.

In addition, a system according to the present invention is easy for developers and system administrators to understand. Once the basic architecture of the system is understood, a straightforward review of the relationships between agents will make it clear what functions and features are invoked in a communication session.

Each policy **120** consists of a policy identification (ID) **122** and a specific instance of a policy **124** as illustrated in **Figure 4**. Policy ID **122** points to a specific policy **124** by employing a pointer **128**. Each specific policy **124** can point to a further policy **120** in ~~the~~ a policy chain ~~126~~ through use of a pointer **129** pointing to the policy ID **122** of the further policy **120**.

A specific example of policies **120** defining the relationship **130** between two agents **110** is illustrated in **Figure 5**. In the example, "222" is a destination and could, for example, represent a telephone number of a person, Fred. A destination agent **112A** representing destination "222" is linked to a node agent **114A** representing Fred through a policy chain of policies **120A** and **120B**. Policy **120A** consists of policy ID **122A**, which has type equal to Day of Week and ID (i.e. the address of a specific Day of Week policy) equal to X. In other words, policy ID **122A** points to specific Day of Week policy X **124A**. As suggested by its policy type, specific policy X **124A** points, via pointer **129A**, to different objects depending on the day of the week. In the example, policy **120B** corresponds with the day of the week of the call. However, if the call occurred on a different day of the week then another policy such as policy **120C** could have been selected.

Policy **120B** consists of policy ID **122B** which identifies the policy type as "node" and the specific policy as node policy Y **124B**. Specific policy Y **124B** points to node agent **114A** representing Fred. The meaning of the relationship between the destination agent **112A** for destination "222" and the

node agent **114A** representing Fred is that if "222" is dialed on certain days of the week then the call will be connected to Fred. Of course, Fred is a person and cannot be contacted directly. Further agents will link the node agent **114A** representing Fred to a suitable device. This will be illustrated in detail later.

~~Figure 6 illustrates~~Figures 6A and 6B illustrate different types of policies. ~~Table 1 in~~ Figure 6A lists ~~a few~~ representative examples of policy types but many others could exist. A particular policy type included in ~~Table 1~~Figure 6A is the group policy. Group policies are useful for logically organizing individuals. For example, individuals who are designated to respond to user requests about technical matters could be the members of a help desk group.

Policies ~~120~~ represent relationships ~~130~~ between agents ~~110~~ at different layers, but policies can also exist within policies. If we think of policies as rules for branching between agents of different layers then policies inside policies add refinements or sub-rules to the branching rules work.

In the example of ~~Table 2 of~~ Figure 6B, a policy inside a policy mechanism, i.e. a selection policy within the group policy, is used to select an individual from that group to respond to the call at hand. This would clearly be useful in distributing work to the members of our help desk group of ~~in~~ our earlier example. Different selection policies use different strategies for passing on the call. A terminal selection policy starts at the head of the list and passes the call to the first available person implementing a hierarchy or priority system of distributing calls. A circular selection more evenly distributes calls than a terminal selection policy by beginning to hunt where the last hunt left off. A broadcast policy passes the communication to everyone at the same time, for example, when a voice mail message is forwarded to everyone in the group. A longest idle selection policy begins by selecting the group member who has been idle the longest.

Another example of a policy inside a policy is a destination policy within a node policy. For instance, different telephone numbers could result in a communication path ending up at the same node but the different calls may continue along different paths out of the node. Accordingly, the node policy points to a different place, i.e. chooses a different policy path, depending on the desired destination. This provides another example of the flexibility inherent in the present system that is unavailable in conventional call control systems. To take a specific example, suppose that the manager of a help group has an assistant who receives calls to the manager to help screen them. On a particular day, the help group is short staffed and the manager agrees to take help desk calls, but still wants the assistant to screen regular calls. In a conventional system, the manager's number would be entered into the help desk group with the result that calls would first go through the assistant. Of course, inline code could be used to make

an exception to the usual call processing but this is awkward and exemplary of a patchwork approach typical of conventional systems. By contrast, a system that uses the present invention can easily implement such an arrangement by noting that the destination of the call is the help desk and distinguishing this call from calls to the manager. Accordingly, calls to the help desk routed to the manager go directly to the manager's phone whereas calls to the manager go to the assistant. The ease with which the system accommodates such changes is a direct consequence of the more powerful architecture of the system, using the layered agent approach.

A key aspect of the system is the use of a database to store information about the system. Entries within tables of the database represent objects, including agents and policies, within the system. The database is used to configure the communication server upon start up. Changes to the objects can be entered into the database, preferably using a graphical user interface. Additionally, any changes made directly to the system will also be recorded in the database so that no information is lost the next time the system starts up or is reconfigured.

**Figure 7** illustrates tables **144**, **146** and **148** which correspond to the destination, the node layer, and the device layer, respectively. Generally, entries or records **142** in a table correspond to objects. Thus in **Figure 7**, the records **142** of the destination table **144** correspond to destination agents **112**, the records **142** of the node table **146** correspond to node agents **114** and the records **142** of the device table **148** correspond to device agents **116**. Similarly, the records **142** of different policy tables **150** correspond to the policies **120** of the system.

In addition, the interrelationships between records ~~142~~ in the database correspond to those between objects. This is shown in the example of **Figure 8** in which records ~~142~~ in the destination table **144A** point to records ~~142~~ in a Time of Day policy table **150A** or a Day of Week policy table **150B**. This is directly analogous to the example of **Figure 5** in which destination agent **112A** for destination "222" points to Day of Week policy **120A**. Recall that Day of Week policy **120A** consists of policy ID **122A** and specific policy X **124A**. Similarly, a record ~~142~~ in destination table **144A** specifies a policy type (indicating the type of table) and a policy ID (indicating which record in the table). From policy tables **150A**, **150B**, further policies **150** can be specified until a node table **146** is reached. Then additional policy tables **150** are ~~possible~~ <u>accessed</u> until ~~the~~ <u>a</u> device table **148** is reached.

A user **220**, or system administrator, can make changes to the system by entering these changes in the database. The user can enter the changes by means of a data entry station **210** as illustrated in **Figure 9**. The information **230A** entered into the data entry station **210** is used to update the database within the

database server **200**, preferably over the WAN or LAN 500, as illustrated by arrow **230B**. Referring also to Figure 10, when~~When~~ the communication server **100** is next brought up or initiated by a user **220**, it receives the data **230C** from the database server **200**, again over the WAN or LAN 500, and creates the appropriate objects (agents and policies). ~~See, for example, Figure 10.~~

**Figure 11** illustrates that a user **220** can update the system, over the network 500, using either the data entry station **210** or a device such as a telephone **410** connected to a phone server **400**. The changes are entered into the database in database server **200** and then updated in the communications server **100**. Note, however, a buffer **160** can be used to hold changes to avoid disrupting existing communications.

Referring to **Figures 12** and 13, there are illustrated examples of user interfaces ~~illustrates an example user interface~~ for entering changes to the database at ~~the~~ a database entry station~~-210~~. Icons corresponding to destinations, persons or network nodes, devices, and policies are available for manipulation in a default configuration. Specifically, there are icons **212** representing destinations, icons **214** representing policies, icons **216** representing nodes, icons **218** representing additional policies and icons **219** representing devices. Unnecessary icons can be removed or additional icons can be added. These icons are organized in rows, somewhat analogous to the layers of the different agents. In order to establish a desired configuration the user uses the interface to connect these icons defining possible communication paths (by the choice of icons representing agents) and defining properties associated with those communication paths (by the choice of policies).

~~An example configuration is shown in~~ **Figure 13** ~~which~~ illustrates a more complex web of destination, policy, node and device connections through the layers. Note that policies can connect to other policies forming a policy chain.. It is also possible to have a direct connection without an intermediary policy if no processing logic is required, for example, if a person always wants to be reached by one particular phone such as a cellular telephone.

A more specific example is given in **Figure 14** in which a particular phone number (i.e. destination) "222" needs to be routed differently depending on the day of the week. In the example, the phone number of the help desk is "222". In this example, the number "222", represented by destination agent **112**, is the phone number of a help desk which is staffed by two groups. Group 1, represented by policy **120D**, is responsible for responding during the week and Group 2, represented by policy ~~**120B**~~**120E**, works weekends. John, represented by node agent **114C**, works seven days a week and thus belongs to both groups. John works in the office during the week but prefers to work from home during the weekend. **Figure 14** illustrates how such a configuration is easily arranged using the objects of the

present invention.

The destination agent 112 ~~corresponding with~~receives a call having destination "222". Destination agent 112 then invokes a ~~points to~~ Day of Week policy 120 which, in turn, invokes ~~points to~~ Group 1 policy 120C on Monday to Friday and Group 2 policy 120D on Saturday and Sunday, based on, for example, system date and time data. Group 1 has a default terminal hunt type starting with Fred, represented by node agent 114A. Accordingly, Fred always gets a help desk call during the week if he is available, otherwise the call passes ~~onto~~ to George, represented by node agent 114B, and then to John at node 114C. This terminal hunt strategy is used to implement a hierarchy or priority scheme in which Fred, at the head of the list, is responsible for most calls, with George and John acting to handle overflow or backup. No intermediate policy is required for Fred and Phone 1, represented by device agent 116A, or George and Phone 2, represented by device agent 116B, since Fred and George are available if and only if they are by their office telephones. For John, however, ~~the~~ a further instance of Day of Week policy 120 is used to determine whether his office phone (Phone 3, represented by device agent 114C) or his home office phone (external number "555-5555" represented by an external number agent 118A) is the appropriate device to reach him, depending on the day of the week.

Calls during the weekend are routed to Group 2 by the first instance of Day of Week policy 120. In this example Group 2 uses a circular hunt type. Thus if the last call was received by Nancy, represented by node agent 114D, and terminating at Phone 4, represented by device agent 116D, then the system will try Mary, represented by node agent 114E, and terminating at Phone 5, represented by device agent 116D, and ~~then~~ finally John at node agent 114C, before passing another call to Nancy at node agent 114D. In contrast to the terminal hunt strategy, this circular hunt strategy is used to equally distribute the work load of responding to calls.

The example of **Figure 14** illustrates an important aspect of the invention. Starting from a first agent, the relationships (defined by policies and structured as a policy chain) associated with that first agent (e.g. destination agent 112) are used to determine a communication path and the second agent (e.g. external number agent 118A) through an intermediate agent (e.g. node agent 114C for John). Note that the policies encountered, starting with destination agent 112 ~~determines~~, determine the communication path and ~~that~~ the second agent is not predetermined. Instead the communication path and the second agent are determined, in this example, by data, such as the day of the week and the state of the system (e.g. whether Fred and George are available).

The calls to John at his external number illustrate a further aspect of the invention. John wants his

calls to go to his home at external number "555-5555". The system administrator wants to ensure that the best route for calling is used, based on system policies. So instead of having a trunk hooked up to John directly at his home, the system finds an appropriate line. ~~Referring to~~ **Figures 15A** to **15C** illustrate how an outbound line is selected. **Figure 15A** shows~~, when the~~ node agent **114C** relaying a call to the external number agent **118A** associated with John's home. As shown in **Figure 15B**, in making this connection, the external number agent **118A** effectively becomes an instance of a destination agent **112X**, which is associated with the ~~for John has selected an~~ external number. ~~the~~ The destination agent **112X~~B~~**, representing ~~that~~ the external number, ~~is asked to provide~~ determines a device ~~for this~~ to call using ~~,~~ ~~Intermediate~~ intermediate policies and nodes. The policies and nodes are used to provide a connection that is consistent with the priorities and preferences of the system. A device, represented by device agent **116G**, is then selected and connected to John's agent **114C**, ~~is notified of the selection and,~~ ~~for~~ For the duration of the call, John's node agent **114C** is associated with ~~this~~ device agent **116G**. As illustrated in **Figure 15C**, the connection between John's agent **114C** and the device agent **116G** is logically seamless.

A second example of a configuration is provided in **Figure 16** which shows the paths for two destinations "333" and "444" represented by destination ~~nodes~~ agents **112B** and **112C**, respectively. Both of these destinations point to ~~a~~ Joe's node agent **114F** ~~representing Joe~~. Joe's node agent **114F** ~~points to~~ invokes one of two group policies, Group policy X **120D** and Group policy Y **120E**. Group policy X **120D** has a terminal continuous hunt policy. Group policy Y 120E has a ~~with their internal hunt policies set to~~ terminal ~~continuous and~~ broadcast~~, respectively~~ policy. Calls ~~corresponding to~~ whose destination data correspond to "333" ~~go to~~ are handled according to the policies of Group X **120D**. ~~whereas calls corresponding to~~ Calls having destination data corresponding to "444" are handled according to the policies of ~~go to~~ Group Y **120E**. Group policy X **120D** contains rules determining the selection of Phones 1 and 2, respectively represented by device agents **116A** and **116B**~~, and~~. Group policy Y **120E** contains rules determining the selection of Phones 2 and 3, respectively represented by device agents **116B** and **116C**. Accordingly, if "333" is called, Phone 1 will ring followed by Phone 2 if there is no answer at Phone 1. If "444" is called then both Phones 2 and 3 will ring.

Generally ~~To~~ to establish a call, a ~~minimum of~~ six agents **110** are ~~generally~~ involved. **Figure 17** shows that a call is split into two sides or half calls **180**. Each half call **180** has at least one device agent **116**, one node agent **114** and one destination agent **112**. There is also a communication record **170** associated with each half call **180** that tracks all information associated with that half call **180** and is discarded after the half call terminates. This is a typical half call model used in many call control systems.

When a user originates a call using a device, a policy determines which node and destination to use. Specifically, the policy determines which path should be chosen. The example of **Figure 18** shows two destination agents instances **112E and 112F** ~~and~~ each connected to two node agents instances **114M and 114N** which are both connected to one instance of a device agent **116**. When this ~~device~~ call originates, it has up to four possible paths to follow. An instance of policy **120** ~~set in the~~ is invoked by device agent **116** ~~will~~ to determine which path to take. The policy can be set statically, ~~i.e.~~e.g. programmed into the database, or dynamically, ~~i.e.~~e.g. by having the user choose which path is taken at the moment the call is made. In addition, the policy can ~~be used to~~ determine whether ~~a~~ static ~~policy~~ or a dynamic ~~policy~~ routing is chosen. ~~In Figure 18, an example static policy 120D always selects the path corresponding with Node Agent 1 and Destination Agent 2.~~

Features are also very important ~~elements in a~~ to communication systems. **Figures 19A** to **19C** show three types of features. **Figure 19A** illustrates a half call, as shown in **Figure 17**, where a first half call **180** is initiated by device agent **116**, and is processed by node agent **114** and destination agent **112**. In the second half call **180**, the destination agent **112** and node agent **114** terminate the call at device agent **116**. ~~Referring to Figure 19A, t~~Type 1 features, such as call swap and conference call, relate to the manipulation or selection of a destination or connection **182** between two half calls **180**. ~~They are in-call call features such as swap and conference.~~ As illustrated in **Figure** ~~**2**~~ **19B**, type 2 features, such as "Do Not Disturb" and "Restriction List", relate to setting up or updating data **162** in or associated with ~~specific~~ an agents **110**. ~~Examples of these type 2 data modifying features include "Do Not Disturb" and "Restriction List".~~ Type 3 features ~~are~~ include advanced programmable system (APS) features, such as Dual Tone Multiple Frequency (DTMF) tones including intelligent tones, or the use of interactive voice response (IVR) technology to replace a tone with another tone or a message. Type 3 features ~~that~~ give feedback to a user **220** who can then make selections. ~~Intelligent tones or the use of interactive voice response (IVR) technology to replace a tone with a message are examples.~~ Features **190** ~~of the system~~ can also be elements of the system that are implemented as objects that can modify the communication path, its properties and the ~~second~~ agent at which the communication path terminates. As illustrated in **Figure 19B**, features can provide data **162** for use by agents **110**.

Further examples of ~~each~~ type 1, 2 and 3 ~~of~~ features are ~~provided~~ listed in **Figures 20A, 20B and 20C** respectively. The examples of **Figure 20** are not exhaustive and additional features are possible. The focus of the present examples is on the use of agents and polices by a system to increase functionality and flexibility. Greater sophistication can, of course, be introduced by adding more intelligence and

functionality to the agents or policies. For example, APS features can be used to get more information from callers using features in order to give allow agents to do more. More specifically, a caller to a service provider may be asked to provide an identifier such as an account number through IVR. Depending on the account number, the call may be routed to different attendants or services. For example, if the service provider is an insurance company, then the account number will identify the name of the customer and that will be used to direct the call to the person responsible for that person's account. As another example, the service provider may be a credit card company offering different levels of service (silver, gold, platinum, etc.) to different card holders. The caller's account number can then be used to route the caller to an appropriate queue or otherwise provide an appropriate level of service.

Features can also ~~use~~ invoke policies. The data that a feature ~~uses~~ provides can be chosen based on policy. This is illustrated in **Figure 21.** ~~When a~~A feature **190** retrieves the data **196** it needs~~,~~. This~~it~~ can be done ~~do this~~ through ~~a set of~~ one or more policies **120**. The policies ~~which then~~ determine which data **196** is ~~used~~ retrieved. An example of ~~this~~ such a feature is Time of Day Call Forwarding~~,~~. Time of Day Call Forwarding~~where the destination that the call forward~~ uses ~~is determined by~~ a specific Time of Day policy **124** to determine the call destination. The feature **190** determines the specific policy **124** by addressing a particular policy identifier **122**. A Time of Day policy instance **124** may, for example, forward calls to a help desk during office hours and forward calls to ~~but invite the caller to leave a~~ voicemail ~~message~~ system outside of office hours.

Features **190** are triggered or invoked in a particular agent **110** through an event occurring in a particular state. **Figure 22** shows how an instance of a policy can be used to flexibly determine which feature is triggered. When an event Z 600 is ~~received~~ detected by an agent **110**, there is an associated communication record **170** which contains the call state. The agent **110** then consults its trigger table **198**. The trigger table **198** contains a list of states and events, and corresponding policy IDs. ~~The~~ Each policy IDs points to a ~~policies~~ policy **120** that ~~eventually point~~ invokes ~~to~~ a particular feature, such as features **190A** and **190B**. In this way, features **190A** and~~,~~ **190B** can have the same state and event trigger, but through a policy, one can have priority over the other. ~~This~~ The priority of one feature over another can also be based on other policies like Time of Day or Calling ID, etc. ~~This means that we have a new policy type of feature. The ID would point to which feature.~~

APS features are triggered in a similar way. **Figure 23** shows how a policy can be used to flexibly determine which APS feature is triggered. When a specific type of event, ~~i.e.~~e.g. tone T, occurs, ~~is given by an~~ agent **110**, through in-line code, ~~then a~~ consults trigger table **198** ~~is consulted~~. This trigger

table **198** contains a list of associated parameters, specifically, tones and reasons and corresponding policy IDs. ~~The~~ Each policy ID~~s~~ points to ~~policies~~ a policy **120** which ~~eventually point to~~ invokes appropriate APS features, such as APS Feature V **191A**. For example when giving a dial tone to start a call, an APS feature may be triggered to select a line. Trigger tables also reside in the database and can be uniquely assigned to an agent.

New policies can be added very simply in this system. The architecture provides the building blocks to create ~~any sort of~~routing, handling and features needed by customers in a communication system. ~~needed by customers, based on their needs and organization.~~